Módulo nº 3

Variáveis escalares: IA32 vs. MIPS32

1. Introdução

Pretende-se com esta aula prática que os alunos identifiquem algumas diferenças fundamentais ao nível das operações aritméticas e passagem de parâmetros entre 2 arquitecturas com filosofias diferentes: IA32 e MIPS32.

Para atingir este objectivo o aluno deve desenvolver um pequeno programa em C, compilá-lo, interpretar o código gerado e alterá-lo.

2. Linguagem de alto nível

Escreva em C, usando o editor de texto que considerar mais adequado, o seguinte programa:

prog.c

```
include <stdio.h>;

int main ()
{
      int i=10,j=5,k,l;

      k = i+j;
      l = k*3;

      printf ("i=%d\tj=%d\n", i, j);
      printf ("k=%d\tl=%d\n", k, l);
}
```

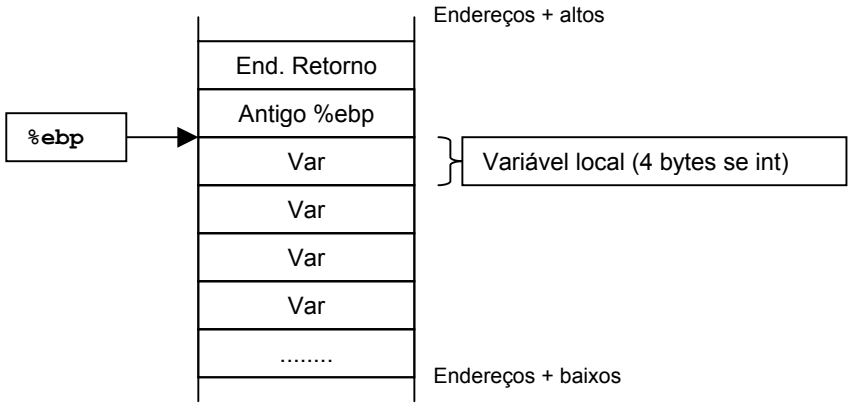
Crie 2 subdirectorias (IA32 e MIPS32) e copie este ficheiro para cada uma delas.

3. Compilação IA32

Na directoria IA32 compile o programa `prog.c` usando o comando

`gcc –S prog.c`

Sabendo que as variáveis locais a cada função são, na arquitectura IA32, geralmente guardadas na *stack*, numa estrutura denominada *activation record* (ver figura seguinte) e que o registo **`%ebp`** é usado como apontador para esta estrutura, usando deslocamentos fixos para aceder a estas variáveis, responda às questões que lhe são colocadas.



Questão 1 – Identifique os deslocamentos relativos a **`%ebp`** correspondentes a cada uma das variáveis locais `i, j, k, l`.

Questão 2 – Identifique a instrução responsável pela operação `k=i+j`.

Questão 3 – Identifique as instruções responsáveis pela operação `l=k*3`.

Questão 4 – Qual é o mecanismo de passagem de parâmetros utilizados pela arquitectura IA32?

Questão 5 – Recompile o seu programa usando o *switch* –O2.Comente as principais diferenças encontradas no código.

Questão 6 – Voltando a recompilar o código em C sem optimização, modifique o código *assembly* de forma a que as operações realizadas sejam `l=i-j; k=l*4;`. Gere o executável usando o comando

`gcc –o prog prog.s`

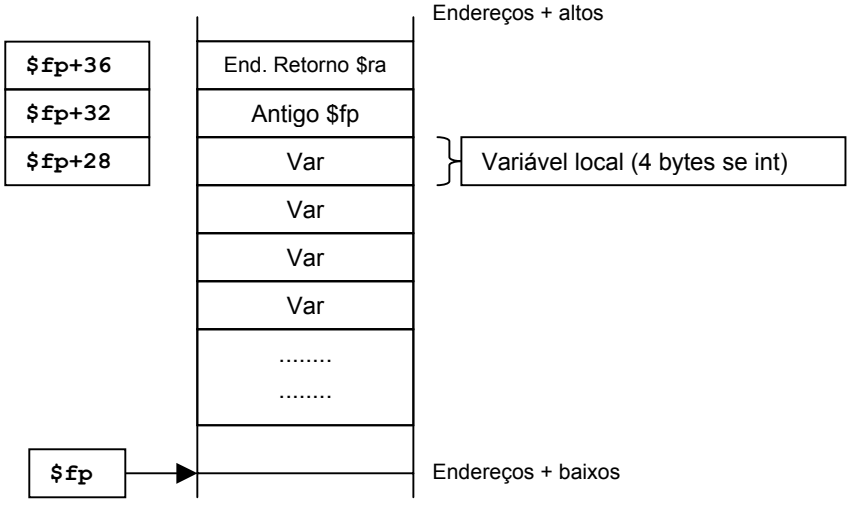
e verifique os resultados executando-o.

4. Compilação MIPS32

Na directoria MIPS32 compile o programa `prog.c` usando o comando

`mips-gcc –mrnames –S prog.c`

Sabendo que as variáveis locais a cada função são, usando este nível de optimização do **`gcc`**, guardadas na *stack*, numa estrutura denominada *activation record* (ver figura seguinte) e que o registo **`$fp`** é usado como apontador para esta estrutura, usando deslocamentos fixos para aceder a estas variáveis, responda às questões que lhe são colocadas.



Questão 7 – Identifique os deslocamentos relativos a **`$fp`** correspondentes a cada uma das variáveis locais `i, j, k, l`.

Questão 8 – Identifique a instrução responsável pela operação `k=i+j`.

Questão 9 – Identifique as instruções responsáveis pela operação `l=k*3`.

Questão 10 – Qual é o mecanismo de passagem de parâmetros utilizados pela arquitectura MIPS32?

Questão 11 – Recompile o seu programa usando o *switch* –O2.Comente as principais diferenças encontradas no código.

Questão 12 – Sabendo que o MIPS32 tem 32 registos de uso geral, pensa que se justifica a utilização da *stack* para guardar as variáveis locais à função?

Questão 13 – Revendo as respostas dadas às questões 2, 3, 4, 8, 9, 10 e 12 comente as grandes diferenças encontradas entre as arquitecturas MIPS32 e IA32.